The HP-75 "Earthquake" Clock Bug: Cause and Solution

Once in a while, the HP-75's TIME and TIME$ commands return an incorrect value, off by as much as one hour. For example, TIME$ might return "9:40:00" when in fact the time is 9:50:00 (off by ten minutes in this example).

The misread is due to the fact that reading the TIME or TIME$ is a process that takes many clock cycles (as each digit of the hardware clock is read one by one into RAM), but the updating of the hardware clock itself appears to be instantaneous to the operating system. This makes it possible for the hardware clock to "tick" (advance) in the middle of the process of reading the TIME or TIME$, making the digits that were read out of the clock *before* the tick no longer valid.

Analogy: Imagine that you are trying to write down the time from a digital clock on the wall. You look up at the clock and see that the hour is 9. You look down at your paper, and write "9". Now you look up and see that the minutes on the clock are 00, so you write down ":00" after the 9. The final result on your paper is "9:00". But you look up immediately afterwards and are surprised to see that the clock now says "10:00". The time you wrote down is one whole hour off. How did this happen?

It happened because at the moment that you first looked at the clock, it said "9:59". You looked at *only* the hour, saw the 9, and wrote it down. *While* you were writing it down (and not watching the clock), it "ticked" forward to 10:00. Then you looked up and read *only* the minutes (now 00) and then wrote them down. Thus you combined the "9" from 9:59 with the "00" from 10:00, to get the incorrect reading "9:00".

Here's a step-by-step illustration of the example given above.

	H	H	:	M	M	
First digit of clock is read	0	9	:	5	9	
Second digit is read	0	9	:	5	9	
Clock "ticks" to next second	1	0	:	0	0	← the "earthquake"
Third digit is read	1	0	:	0	0	
Fourth digit is read	1	0	:	0	0	

Final result: **09:00**, even though the time is actually 10:00.

Somebody at HP compared this phenomenon to the difficulty of walking in a straight line during an earthquake. Ordinarily walking in a straight line is easy, because the earth doesn't move, so you always know where your next footstep will land. But during an earthquake, terra firma suddenly stops being firma, making your next step land somewhere you didn't expect, resulting in a jagged path. That's

what happens during a "clock earthquake": while we are stepping across the digits of the clock, they unexpectedly move beneath us, resulting in a jagged clock reading.

The successor to the HP-75 was the HP-71B. Its design team (who also referred to a time quake as a "kerchunk") avoided this bug in the HP-71B by reading the clock *twice* every time the TIME or TIME$ command was executed. If the result was the same, it was returned as-is. However, if the two results were different, then the "earthquake" must have happened during one of the two reads, so the clock was read a third time (before the next earthquake could occur) and its result was returned.

Even though the HP-71B's hardware clock counts *backwards*, and the CPU reads the clock into RAM backwards too (least significant digit first, most significant digit last), it doesn't matter, because this bugfix works either way. Here's how it is documented in the HP-71B Internal Design Specification source code for the GETTIM (Get Time) entry point (address 1254A):

512 hz countdown timer operates asynchronously of CPU clock. It can decrement at any time, including during a read. A double-read algorithm assures that we do not encounter situations such as follows:
Timer = 120000, decrements after CPU reads first three digits, yielding a reading of 11F000 (since CPU reads LSM to MSN).

Algorithm:
Read timer -> B.
Read timer -> A.
If A#B then read timer ->A.
[Return A]

[Credit: The above source code documentation for GETTIM was written by "NM", whoever that is. It is dated 3 June 1982.]

-jkh- 2/2/2014